HYDROMORPHO: A COUPLED MODEL FOR UNSTEADY STOKES/EXNER EQUATIONS AND NUMERICAL RESULTS WITH FEEL++ LIBRARY∗,**

Nora Aïssiouene1, Tarik Amtout2, Matthieu Brachet3, Emmanuel Frenod4, Romain Hild5, Christophe Prud’homme6, Antoine Rousseau7 and Stephanie Salmon8

Abstract. We propose to couple the Exner equation with the Stokes equations to model the sediment transport for bed load in geophysical flow. This work is a preliminary in context study to directly model the hydrodynamic flow by the unsteady Stokes equation instead of the classical shallow water equation. We focus in this proceeding to the coupling applying fluid structure interaction approach to morphodynamical behavior. In other words, we follow the approach of fluid interaction models replacing the structure equation by the Exner equation. The aim of this work is to validate the procedure proposed. These equations are solved by finite element method using the library FEEL++ (REF).

Résumé. ...

INTRODUCTION

In hydrodynamic field, many studies have been done to understand and predict the dynamical of sediments at the bottom of flow which is a significant and complex process for many geophysical situations. Morphodynamics modelling is a broad subject whose principles can be found in several texts [23], [24]. We can distinguish two types of sediments transport, the suspended load and the bed load. In this proceeding, we focus on the bed load transport and its impact on the hydrodynamics. The difficulty remains in the necessity to couple the sediment transport models with the hydrodynamic models which can be also complex in function of the application, and then to develop a robust and stable numerical method.

On the one hand, the sediments transport is usually modeled by the classical Exner equation [25] and several laws of transport have been proposed (see [13] to have details of some classical laws) by physical arguments or

∗...

**...

1 Inria - CEREMA - Sorbonne Universités, UPMC Univ Paris 06 - CNRS, UMR 7598, Laboratoire Jacques-Louis Lions
2 Faculté des Sciences et Techniques Tanger, Maroc
3 Institut Elie Cartan de Lorraine, Université de Lorraine, Site de Metz, Bât. A Ile du Saulcy, F-57045 Metz Cedex 1
4 Université Bretagne-Sud, LMBA UMR 6205, Vannes
5 Institut de Recherche Mathématique Avancée, Strasbourg
6 Institut de Recherche Mathématique Avancée, Strasbourg
7...
8...

© EDP Sciences, SMAI 2015
closure relations. On the other hand, models as shallow water equations are used to model the hydrodynamics, and recently in [7,14] a model derived from the Navier-Stokes equations that has an energy balance.

Concerning the numerical methods that have been established for these models, the main numerical schemes are developed for the hyperbolic systems with source terms for the hydrodynamic flow (see REF). Therefore, finite volume scheme are applied for the shallow water system ([1,2,26,27]). The problem lies in the coupling of the numerical schemes. Indeed, in the shallow water models, the topography is a source term and the Exner equation gives the evolution of the bottom in terms of the velocity of the fluid. Then, two strategies are distinguished, the splitting one and the non-splitting one (REF). The splitting methods are easier to implement but generate instabilities in specific situations (see [9]). On the contrary, more complicated models, for instance involving relaxation, have to be used to take into consideration the fully coupled model ([3,18]).

Notice that the shallow water model is based on a hydrostatic assumption. It is deduced from the Navier Stokes equations, neglecting the vertical acceleration. Many other free surface models have been developed to take into account a non hydrostatic pressure ([5,6,21,28]). However, numerical methods are much more complicated than for the hydrostatic model. It is not in the scope of this study to consider this kind of models, but it could be interesting to compare the results obtained with a non-hydrostatic model.

For this study, in order to take into account the non hydrostatic contribution, we chose to work with the unsteady Stokes equations.

The approach of this study is to apply the fluid structure interaction theory (see [8,16]) to the coupled problem Stokes-Exner. We consider the sediments transport as a moving structure and use Stokes equations to model the fluid flow. Then, we use a Arbitrary Lagrangian Eulerian (ALE) method (see [16]) to couple the hydrodynamic flow and the bottom equation and we give a variational framework of the final model in order to apply finite element method.

Notice that the choice of the a finite element method applied to the Navier-Stokes equations implies to have costly numerical simulation, that is why we use a finite element library Feel++ within which parallelism is embedded. This allows to implement the method using variational formulation with high performance computing.

The proceeding is organized as follow, the first part is devoted to the description of the the model of the fluid and the description of the model of the bottom. In a second part, it is explained why a method like the ALE is necessary to couple the models. The third part establishes a complete ALE formulation of the Stokes-Exner model. Then, a variational formulation is given with the different boundary conditions that we explore. Finally, some numerical results are presented to evaluate the model and the solution method.

1. The model

For this preliminary study, we consider the domain

$$\Omega(t) = \{(x,z) \in \mathbb{R}^2 \mid 0 \leq x \leq l, \ b(x,t) \leq z \leq 1\}$$

where \(l > 0\) is the cavity length and \(b(x,t)\) is the bottom topography in \(x\) at time \(t\). We denote the velocity \(u(x,z,t) = (v(x,z,t),w(x,z,t))^T\), and the pressure \(p(x,z,t)\).

We also denote by \(\Gamma(t) = \Gamma_{in}(t) \cup \Gamma_{out}(t) \cup \Gamma_s \cup \Gamma_b(t)\) the boundaries (see Figure 1):
- \(\Gamma_{in}(t) = \{0\} \times [0(b(0,t),1)\]
- \(\Gamma_{out}(t) = \{l\} \times [b(1,t),1]\)
- \(\Gamma_s = [0,l] \times \{1\}\)
- \(\Gamma_b(t) = \{(x,z) \in \mathbb{R}^2 \text{ s.t. } z = b(x,t), x \in [0,l]\}\)
The coupled model leads to solving the non-steady Stokes problem in the fluid domain $\Omega(t)$ and the Exner equation to give the boundary $\Gamma_b(t)$.

1.1. Hydrodynamical Model

We consider the unsteady Stokes problem on the domain $\Omega(t)$

$$\frac{\partial \mathbf{u}}{\partial t} - \mu \Delta \mathbf{u} + \nabla p = 0 \text{ on } \Omega(t),$$

$$\text{div}(\mathbf{u}) = 0 \text{ on } \Omega(t),$$

where $\mathbf{u} = (u, w)^T$ is the velocity of the fluid, $p$ is the pressure and $\mu > 0$ is the viscosity. This problem is completed with the following boundary conditions.

- Dirichlet boundary condition :
  $$\mathbf{u} = \mathbf{g} \text{ on } \Gamma(t)$$

  In particular, we can look at the condition :
  $$\mathbf{u} = 0 \text{ on } \Gamma_{\text{in}}(t) \cup \Gamma_{\text{out}}(t) \cup \Gamma_b(t)$$

  $$\mathbf{u} = (1, 0)^T \text{ on } \Gamma_s$$

  This is the case of the lid driven cavity.

- Mixed Dirichlet/Neumann/slip boundary condition :

  $$\mathbf{u} = \mathbf{g}_1 \text{ on } \Gamma_s$$

  $$\mathbf{p} \mathbf{n} - \mu \frac{\partial \mathbf{u}}{\partial n} = \mathbf{g}_2 \text{ on } \Gamma_{\text{in}}(t) \cup \Gamma_{\text{out}}(t)$$

  $$\mathbf{u} \cdot \mathbf{n} = 0 \text{ on } \Gamma_b(t)$$

  $$\mathbf{p} \mathbf{n} - \mu \frac{\partial \mathbf{u}_t}{\partial n} = \mathbf{g}_3 \text{ on } \Gamma_b(t)$$

The condition (7) impose the force by using Dirichlet condition.
The condition (8) let free the velocity at the inlet and at the outlet.
The condition (9) impose the normal component of the velocity to be null at the bottom. It is the
condition of impermeability of the domain.

The condition (10) let free the tangential component of the velocity at the bottom. It is needed to have a displacement of the bottom.

1.2. Morphodynamic model

The sediment dynamics is based on the formulation of a sediment continuity equation stating that the time variation of the sediment layer in a certain volume is due to the net variation of the solid transport through the boundaries of the volume. The mathematical expression of such law is known as the Exner equation [22] presented in this form:

$$\frac{\partial b_z}{\partial t} + \xi \frac{\partial Q}{\partial x} = 0$$ \text{ sur } [0, l] \tag{11}

where \(b_z(x, t)\) is the bed elevation, \(\xi = \frac{1}{1-p}\) is the material porosity and \(Q\) denote the solid transport discharge along the x coordinate influenced by the velocities \(u\) and \(w\). The formulation of the bed load discharge \(Q\) can be based on deterministic laws ([4], [11], [29]) or in probabilistic methods ([12], [19]), often supported by experimentation. Grass [20] discussed one of the most basic sediment transport laws that in 1D can be written as

$$Q = \alpha |u|^{3/2}$$

where \(\alpha \in \mathbb{R}\) is an empirically parameter and represents the effects associated to the grain size and the kinematic viscosity. Ranging typically from 0 to 1, it represents a stronger interaction between flow and sediment as it approaches.

1.3. Arbitrary Lagrangian Eulerian Method (ALE)

In fluid mechanic, we can enumerate two ways to see the same problem: Lagrangian and Eulerian formulation. Lagrangian formulation is similar to keep track the location of each fluid particles. The velocity \(U\) and the density \(\rho\) depend only of \(x_0\) the initial position of the particles and of \(t\) the time. Then, the time derivative of a quantity \(F\) is given by the total time derivative:

$$\frac{DF}{Dt}.$$  

This method is interesting to understand or fixed the equation but it’s costly to obtain numerical simulations.

For the Eulerian formulaton, the main idea is to fixed a system of coordinate and follow the flux of particles. In this context, the velocity \(u\) and the density \(\rho\) depend of \(x\) the position in a global system of coordinates and \(t\) the time. Now, for a function \(F\), the time derivative is given by:

$$\frac{\partial F}{\partial t} + u \cdot \nabla F$$

The link between the Eulerian and the Lagrangian time derivative is:

$$\frac{D}{Dt} = \frac{\partial}{\partial t} + u \cdot \nabla \tag{12}$$

In this works, we use Eulerian formulation for the fluid equation.

To solve numerically a problem, we must mesh the domain. Each point of this mesh have a coordinate but the domain is time dependant. It’s saying that the mesh at time \(t\) is wrong at time \(t + \Delta t\). Remesh at each time step is too expansive and keep only one mesh is impossible, so we want to move the mesh with the bottom.
This idea is classic in Fluid Structure Interaction: it's the Arbitrary Lagrangian Eulerian method. This method was first developed for finite difference in [?], and scope to finite element methods in [?], and [?]. In 2004, [?] build a method for great order elements.

The main idea is to work in a reference domain \( \hat{\Omega} \) (in the following, we write \( \hat{\cdot} \) all quantities concerning the reference domain) to solve the fluid equation.

In our context, we choose \( \hat{\Omega} \) the rectangular \([0, l] \times [0, 1]\) and the following boundaries:
- \( \hat{\Gamma}_{\text{in}} = \{0\} \times [0, 1] \)
- \( \hat{\Gamma}_s = [0, l] \times \{1\} \)
- \( \hat{\Gamma}_{\text{out}} = \{l\} \times [0, 1] \)
- \( \hat{\Gamma}_b = [0, l] \times \{0\} = \hat{\gamma}_b \times \{0\} \) where \( \hat{\gamma}_b = [0, l] \).

To come back from the reference domain \( \hat{\Omega} \) to the physical domain \( \Omega(t) \), the main tool is the ALE map:

\[
\mathcal{A}^t: \begin{cases} 
\hat{\Omega} 
\mapsto \Omega(t) \\
\hat{x} 
\mapsto x(\hat{x}, t)
\end{cases}
\]  

(13)

We must determine \( \mathcal{A}^t \) for every point of the mesh.

The point \( x(t) \in \Omega(t) \) can be found with:

\[
x(t) = \mathcal{A}^t(\hat{x}) = \hat{x} + \hat{d}_d(\hat{x}, t)
\]  

(14)

where \( \hat{d}_d(\hat{x}, t) \) is the displacement of \( \hat{x} \) between \( \hat{\Omega} \) and \( \Omega(t) \). Notice that \( x(t) \) is time dependant. Then, we can define the mesh velocity:

\[
\hat{w}(\hat{x}, t) = \frac{\partial \mathcal{A}^t}{\partial t}(\hat{x}) = \frac{\partial \hat{d}_d}{\partial t}(x, t)
\]  

(15)

\( \hat{w} \) is from \( \hat{\Omega} \times \mathbb{R}^+ \) to \( \mathbb{R}^d \). So, the equivalent from \( \Omega(t) \times \mathbb{R}^+ \) to \( \mathbb{R}^d \) is \( w = \hat{w} \circ (\mathcal{A}^t)^{-1} \). We must determined the displacement \( \hat{d}_d \) in \( \hat{\Omega} \). \( d_d \) must be smooth. In practice, \( d_d \) is the solution of a PDE like harmonic or Wislow equation. Sometimes, when the displacement is bigger, we can obtain a bad new mesh. Some methods are more effective than other to solve this (see [8] for more details). We often need to transport an equation from \( \hat{\Omega} \) to \( \Omega(t) \) and mutually. Let \( u: \Omega(t) \times \mathbb{R}^+ \rightarrow \mathbb{R}^d \), then the corresponding map in \( \hat{\Omega} \) is \( \hat{u} = u \circ \mathcal{A}^t \). If \( \frac{D u}{D t} \) is the times-derivative of \( u \) in ALE, we have the following equality:

\[
\frac{D u}{D t} = \frac{\partial u}{\partial t} \bigg|_x + w \cdot \nabla u
\]  

(16)

where \( \frac{\partial u}{\partial t} \) is the times partial derivative.

- If \( w = u \), the mesh is moving with the particles so the description is Lagrangian.
- If \( w = 0 \), the mesh don’t move and the description is Eulerian.
1.4. Coupled model

In this part, we focus on the coupled model. We denote by $\hat{x}$ a point in the reference domain $\hat{\Omega}$ and by $\Omega(t)$ the deformed domain after the transformation. The deformation of the mesh leads to consider the Lagrangian derivative $D$ defined by (16) which is also called the ALE derivative where the velocity $w$ is defined as:

$$\hat{w} = \frac{\partial \hat{d}}{\partial t}$$

and represents the velocity of the displacement of the mesh, that is to say the velocity of the particle in the referential domain. This allows to write the fluid model with a moving mesh $\Omega(t)$. Concerning the boundary conditions, we still consider a slip boundary condition at the interface between the topography and the fluid. The following model is still valid with the previous boundary conditions (4), (5)-(6), (7)-(10).

The complete model is composed of the equation of the fluid in two dimensional domain, the equation of the topography in one dimensional domain and the ALE equation in two dimensional domain. After the changing variable (16), the coupled model reads:

1.4.1. Fluid equation

The fluid equation is given by:

$$\frac{D u}{D t} - w \cdot \nabla u - \mu \Delta u + \nabla p = 0 \quad \text{on } \Omega(t)$$

$$\text{div}(u) = 0 \quad \text{on } \Omega(t)$$

$$+ \text{BC}$$

1.4.2. Bottom equation

We consider the one dimensional domain $\hat{\gamma}_b = [0, l]$, on which the bottom topography $\hat{b}(\hat{x}, t) \quad \forall \hat{x} \in \hat{\gamma}_b$ is defined by the Exner equation (11).

$$\frac{\partial \hat{b}_z(x, t)}{\partial t} + \frac{\partial \hat{Q}(\hat{x}, t)}{\partial \hat{x}} = 0 \quad \forall \hat{x} \in \hat{\gamma}_b, t > 0$$

$$\hat{b}_z(\hat{x}, 0) = \hat{b}_z, 0(\hat{x})$$

Using EQREF, the sediment law $\hat{Q}$ is written in the reference domain by:

$$\hat{Q}(\hat{x}, t) = Q(A^t((\hat{x}, 0)^T), t) \quad \forall \hat{x} \in \hat{\gamma}_b$$

$$= \alpha (u_\tau \circ A^t((\hat{x}, 0)^T))^{3/2}$$

1.4.3. Displacement equation

This displacement needs to be extended in the fluid domain to associate a new ALE map over the mesh. In order to do this, we use a harmonic extension.

$$- \Delta \hat{d}_s = 0 \quad \text{on } \hat{\Omega}$$

$$\hat{d}_s = 0 \quad \text{on } \hat{\Gamma}_s$$

$$\frac{\partial \hat{d}_s}{\partial n} = 0 \quad \text{on } \hat{\Gamma}_{in} \cup \hat{\Gamma}_{out}$$

$$\hat{d}_s = (0, \hat{b}_z(\hat{x}, t))^T \quad \text{on } \hat{\Gamma}_b$$

This allows us to have a given displacement defined on $\hat{\Gamma}_b$, to let free the boundaries $\hat{\Gamma}_{in}$ and $\hat{\Gamma}_{out}$, and to fix the boundary $\hat{\Gamma}_s$. The harmonic problem diffuse the displacement $\hat{d}_s$ on all the domain.
1.4.4. Equation of \( w \)

We denote by \( \hat{w} \) the velocity of the displacement

\[
\hat{w}(\hat{x}, t) = \frac{\partial \hat{d}(\hat{x}, t)}{\partial t}
\]  

(28)

Then, using the ALE transformation, we can compute the velocity \( w \) in the domain \( \Omega(t) \):

\[
w(x, t) = \hat{w}((A^t)^{-1}(x), t)
\]  

(29)

2. Variational formulation

This part is devoted to the variational formulation of the problem taking the ALE description into account.

The problem leads to find \( u \in V \) and \( p \in Q \) such that the fluid equation (17) is satisfied. Let \( X \) be the functional set of test functions. Notice that the sets \( V \), \( X \) and \( Q \) will be defined later. In practice, the sets \( V \) and \( X \) can be different, they depend on the boundary conditions. By multiplying (17) with a test function \( v \in V \) and (18) by a test function \( q \in Q \), and then integrating by part, we get:

\[
\int_{\Omega} \frac{D}{Dt} u \cdot v - \int_{\Omega} w \cdot \nabla u \cdot v + \int_{\Omega} \nabla u : \nabla v - \int_{\Omega} \frac{\partial u}{\partial n} \cdot v - \int_{\Gamma} p \text{div}(v) + \int_{\Gamma} p n \cdot v = \int_{\Omega} F \cdot v
\]  

(30)

\[
\int_{\Omega} \text{div}(u) q = 0
\]  

(31)

Then, using the Reynolds transport formula on the first term of (30):

\[
\frac{d}{Dt} \int_{\Omega} u \cdot v - \int_{\Omega} u \nabla \cdot w \cdot v - \int_{\Omega} w \cdot \nabla u \cdot v + \int_{\Omega} \nabla u : \nabla v - \int_{\Omega} \frac{\partial u}{\partial n} \cdot v
\]  

\[
- \int_{\Gamma} p \text{div}(v) + \int_{\Gamma} p n \cdot v = \int_{\Omega} F \cdot v
\]  

(32)

\[
\int_{\Omega} \text{div}(u) q = 0
\]  

(33)

We define the following forms:

\[
a_1(u, v) = \int_{\Omega(t)} \mu \nabla u : \nabla v \, dx \quad \forall u \in V, v \in V
\]  

(34)

\[
a_2(u, v) = -\int_{\Omega} u (\nabla \cdot w) \cdot v - \int_{\Omega} w \cdot \nabla u \cdot v \, dx \quad \forall u \in V, v \in V
\]  

(35)

\[
a(u, v) = a_1(u, v) + a_2(u, v)
\]  

(36)

\[
b(u, q) = \int_{\Omega(t)} q \, \text{div}(u) \, dx \quad \forall u \in V, q \in Q
\]  

(37)

\[
L(v) = \int_{\Omega(t)} F(t) \cdot v \, dx \quad \forall v \in V
\]  

(38)

3. Boundary conditions

In this part we explore various boundary conditions that are imposed to simulate for instance an inflow at the inlet, a free outflow at the outlet or a forcing flow on the top to represent the free surface.

3.1. Dirichlet boundary conditions

In many geophysical tests, it is usual to simulate an inflow at the inlet of the domain by imposing a given velocity in the numerical method. In this part, we give the variational formulation corresponding to this situation
and to simplify, we impose the Dirichlet condition for all the boundaries, which is not a physical situation. We introduce the spaces:

\[ V = \{ u \in H^1(\Omega(t)), \quad u = g \text{ on } \Gamma \} \]

\[ X = H^1_0(\Omega) = \{ v \in H^1(\Omega(t)), \quad v|_{\Gamma} = 0 \} \]

and

\[ Q = L^2_0(\Omega(t)) = \{ q \in L^2(\Omega(t)), \quad \int_{\Omega(t)} q \, dx = 0 \} \]

So, for the boundary conditions (4), the problem writes:

Find \( u \in V, \, p \in Q \) such that

\[
\frac{d}{dt} \int_{\Omega} u \cdot v - \int_{\Omega} \nabla u \cdot \nabla v - \int_{\Omega} \nabla u : \nabla v - \int_{\Omega} p \, \text{div}(v) = \int_{\Omega} F \cdot v \quad \forall v \in X \tag{39}
\]

\[
\int_{\Omega} \text{div}(u) q = 0 \quad \forall q \in Q \tag{40}
\]

With the notations (34)- (37), the problem writes :

Find \( u \in V, \, p \in Q \) such that :

\[
\frac{d}{dt} \int_{\Omega} u \cdot v - a(u, v) + b(v, q) = L(v) \quad \forall v \in X \tag{41}
\]

\[
b(u, q) = 0 \quad \forall q \in Q \tag{42}
\]

### 3.2. Mixed Neumann/Dirichlet boundary conditions

In this part, we consider Dirichlet condition on the surface and we specify the boundary conditions that we impose on the inlet (resp. outlet) to simulate a free inflow (resp. outflow). This leads to impose a Neumann boundary conditions on the Stokes equations, which are the natural ones. Notice that in order to not treat the interface between the topography and the fluid in this part, we impose a Dirichlet boundary condition on the bottom boundary. We consider the problem (2)-(3) with the following boundary conditions:

\[ u = g_1 \quad \text{on } \Gamma_s \cup \Gamma_b, \] \[ pn - \mu \frac{\partial u}{\partial n} = g_2 \quad \text{on } \Gamma_{in} \cup \Gamma_{out}. \] \[ \tag{43} \tag{44} \]

and we define the spaces

\[ V = \{ u \in H^1(\Omega(t)), \quad u = g \text{ on } \Gamma_s \cup \Gamma_b \}, \]

\[ X = \{ u \in H^1(\Omega(t)), \quad u = 0 \text{ on } \Gamma_s \cup \Gamma_b \}, \]

\[ Q = L^2_0(\Omega(t)). \]

Then the boundary terms in (30) writes

\[
\int_{\Gamma_{in} \cup \Gamma_{out}} \left( pn - \mu \frac{\partial u}{\partial n} \right) \cdot v = \int_{\Gamma_{in} \cup \Gamma_{out}} g_2 \cdot v.
\]
where $g_2$ is a given function. Then, the problem writes:

Find $u \in V$, $p \in Q$ such that:

$$\frac{d}{dt} \int_{\Omega} u \cdot v + a(u, \varphi) + b(v, q) = L(v) \quad \forall v \in X$$

(45)

$$b(u, q) = 0 \quad \forall q \in Q$$

(46)

The bilinear form $a$ and $b$ are defined by (36)-(37), and $L$ is defined by:

$$L(v) = \int_{\Omega(t)} F(t) \cdot v + \int_{\Gamma_{in} \cup \Gamma_{out}} g_2 \cdot v$$

(47)

### 3.3. Slip boundary conditions

In this section, we are interested in the interface between the fluid and the topography and we preconise to have a slip boundary condition on $\Gamma_b$, which is physically consistent with the sediments transport model chosen in this study, namely the bed load transport.

Then, we give the variational formulation with slip boundary condition $u \cdot n = 0$ on $\Gamma_b$. To this aim, we rewrite the problem:

Find $u \in V, p \in Q$

$$\frac{d}{dt} \int_{\Omega} u \cdot v - \int_{\Omega} u \nabla \cdot w \cdot v - \int_{\Omega} w \cdot \nabla u \cdot v$$

$$+ \int_{\Omega} \nabla u : \nabla v - \int_{\Omega} p \text{div}(v) + \int_{\Gamma} \left( \frac{\rho n - \mu \frac{\partial u}{\partial n}}{\mu} \right) \cdot v = \int_{\Omega} F \cdot v$$

$$\forall v \in X$$

(48)

$$\int_{\Omega} \text{div}(u) q = 0 \quad \forall q \in Q.$$  

(49)

For the sake of clarity, we preconise a Dirichlet boundary conditions on the velocity for the boundaries $\Gamma_{in}, \Gamma_{out}$ and $\Gamma_s$, then the terms at boundaries vanish in the variational formulation and we focus on the on the interface $\Gamma_b$ between the fluid and the topography. It is not natural to impose a slip boundary condition in the Stokes problem, and this problem has been widely studied to having a well posed problem with this boundary condition.

- A first strategy, studied in [10], consists in giving a condition on the stress tensor $\sigma \tau = g$ defined by:

$$\sigma = (\mu \nabla u - p \delta d)$$

(50)

We rewrite the test function $v = (v \cdot n) n + (v \cdot \tau) \tau$ where $n$ is the normal component and $\tau$ is the tangential component on $\Gamma_b$. Taking $g = 0$ and

$$v \in V = \{ v \in H^1(\Omega)^2, v \cdot n|_{\Gamma_b} = 0 \},$$

(51)

it straightforward to verify that the variational formulation writes

$$\frac{d}{dt} \int_{\Omega} u \cdot v - \int_{\Omega} u \nabla \cdot w \cdot v - \int_{\Omega} w \cdot \nabla u \cdot v + \int_{\Omega} \nabla u : \nabla v - \int_{\Omega} p \text{div}(v) = \int_{\Omega} F \cdot v$$

(52)

$$\int_{\Omega} \text{div}(u) q = 0.$$  

(53)

- A second strategy consist in giving a condition on the velocity at the boundary. To this aim, we notice that

$$\int_{\Gamma_b} \left( \frac{\rho n - \mu \frac{\partial u}{\partial n}}{\mu} \right) \cdot v = \int_{\Gamma_b} \sigma n (v \cdot n) - \int_{\Gamma_b} \mu \frac{\partial u}{\partial n} \cdot (v \cdot \tau)$$

(54)
We rewrite the test function $v$ in terms of the normal component and the tangential component, as for the previous case. The problem writes:

$$
\frac{d}{dt} \int_\Omega u \cdot v - \int_\Omega u \nabla \cdot w \cdot v - \int_\Omega w \cdot \nabla u \cdot v \\
+ \int_\Omega \nabla u : \nabla v - \int_\Omega p \, \text{div}(v) \\
+ \int_{\Gamma_b} \sigma n(v \cdot n) - \int_{\Gamma_b} \mu \frac{\partial u}{\partial n} \cdot (v \cdot \tau) = \int_\Omega F \cdot v
$$

(55)

$$
\int_\Omega \text{div}(u) q = 0
$$

(56)

Using the relation (54) and imposing $\frac{\partial u}{\partial n} \cdot \tau + \alpha (u \cdot \tau)\tau = g$ with $\alpha > 0$ (see [17]), the variational formulation writes:

$$
\frac{d}{dt} \int_\Omega u \cdot v - \int_\Omega u \nabla \cdot w \cdot v - \int_\Omega w \cdot \nabla u \cdot v
$$

(57)

$$
+ \int_\Omega \nabla u : \nabla v - \int_\Omega p \, \text{div}(v) \int_\Gamma (\alpha (u \cdot \tau)(v \cdot \tau)) = \int_\Omega F \cdot v + \int_\Gamma g(v \cdot \tau)
$$

(58)

$$
\int_\Omega \text{div}(u) q = 0
$$

(59)

• An other alternative leads to using a penalty method. As for the previous case, we take $v \in V = \{ v \in H^1(\Omega)^2 \, , \, v \cdot n_{\Gamma_b} = 0 \}$. In order to impose the condition $u \cdot n = 0$ for the velocity which is not natural in the variational formulation, we consider the formulation (48) and penalize the natural boundary condition :

$$
\left( p n - \mu \frac{\partial u}{\partial n} \right)_{\Gamma_b} = \frac{1}{\epsilon}(u \cdot n)n
$$

where $\epsilon \ll 1$. The variational formulation becomes:

Find $u \in V$ and $p \in Q$ such that:

$$
\frac{d}{dt} \int_\Omega u \cdot v + a(u, v) + b(v, p) = L(v) \quad \forall v \in V
$$

(60)

$$
\begin{align*}
 b(u, q) &= 0 \quad \forall q \in Q
\end{align*}
$$

(61)

with the bilinear form $b$ defined by (37), $L$ defined by (47) and $a$ defined by

$$
a(u, v) = a_1(u, v) + a_2(u, v) + \frac{1}{\epsilon} \int_{\Gamma_b} (u \cdot n)(v \cdot n) \, d\sigma
$$

(62)

It is proved by Dione in [10] that this problem converges to the problem with slip boundary conditions when $\epsilon$ tends to zero.

3.4. Variational formulation of the Exner equation

As for the fluid equation, a variational formulation of the the Exner equation is given for the complete method. Taking a test function $\phi \in L^2(\gamma_b)$, it gives:

$$
\frac{\partial}{\partial t} \int_{\gamma_b} b_2(\dot{x}, t) \phi(\dot{x}) \, d\dot{x} + \int_{\gamma_b} \frac{\partial \tilde{Q}(\dot{x}, t)}{\partial \dot{x}} \phi(\dot{x}) \, d\dot{x} = 0
$$

(63)
Throughout the rest of the document, we use this functional sets.

We choose the following compatible spaces (see [15]):

\[ \mathbf{V}_h = \{ \mathbf{v} \in C^0(\Omega) \text{ s.t. } \mathbf{v}|_K \in P_2^k \text{ for all } K \in \mathcal{T}_h \} \]

\[ Q_h = \{ q \in C^0(\Omega) \text{ s.t. } q|_K \in P_1 \text{ for all } K \in \mathcal{T}_h \} \cap L_0^2(\Omega) \]

Where \( \mathcal{T}_h \) is the set of mesh elements and \( P_k \) is the set of polynomial function of degree \( k \). The space \( \mathbf{V}_h \) can be adjust for specific boundary condition.

Throughout the rest of the document, we use this functionals sets.

### 4. Numerical method

#### 4.1. Discretisation in time

We will now discretise the time derivative with a Backward Differentiation Formula (BDF) of order 1. It correspond to the backward Euler method but we can consider a better method in a future research. Let \( \Delta t \) be the time step, \( t_i = t_0 \) the initial time, \( t_n = n\Delta t \) and \( x^n \) a field at time \( t_n \).

We can then rewrite the variational formulation at time \( t_{n+1} \):

\[
\int_{\Omega} \frac{\mathbf{u}^{n+1} - \mathbf{u}^n}{\Delta t} \cdot \mathbf{\varphi} + a(\mathbf{u}^{n+1}, \mathbf{\varphi}) + b(\mathbf{u}^{n+1}, p^{n+1}) = L(\mathbf{\varphi}) \quad \forall \mathbf{\varphi} \in \mathbf{V} \\
b(\mathbf{u}^{n+1}, q) = 0 \quad \forall q \in Q
\]

(66)

(67)

since \( \mathbf{u}^n \) is known, this leads to :

\[
\int_{\Omega} \frac{1}{\Delta t} \mathbf{u}^{n+1} \cdot \mathbf{\varphi} + a(\mathbf{u}^{n+1}, \mathbf{\varphi}) + b(\mathbf{\varphi}, p^{n+1}) = L(\mathbf{\varphi}) + \int_{\Omega} \frac{1}{\Delta t} \mathbf{u}^n \cdot \mathbf{\varphi} \quad \forall \mathbf{\varphi} \in \mathbf{V} \\
b(\mathbf{u}^{n+1}, q) = 0 \quad \forall q \in Q
\]

(68)

(69)

#### 4.2. Discretisation in space

We consider in this section a subdivided domain \( \mathcal{T}_h \) and the finite dimensional spaces \( \mathbf{V}_h \) and \( Q_h \) which are the discrete spaces of \( \mathbf{V} \) and \( Q \). As well, we approximate \( \mathbf{u} \) and \( p \) by \( \mathbf{u}_h \) and \( p_h \). We use a Galerkin approximation taking the fields \( \mathbf{u}_h \), and \( p_h \) in \( \mathbf{V}_h \) and \( Q_h \), it reads:

\[
\mathbf{u}_h = \sum_{i=1}^N \alpha_i \varphi_i, \quad p_h = \sum_{i=1}^M \beta_i \psi_i
\]

(70)

where we denote \( N = \dim \mathbf{V}_h \) and \( M = \dim Q_h \) and \( \varphi \) (resp. \( \psi \)) is the basis function of \( \mathbf{V}_h \) (resp. \( Q_h \)). Then, the semi discrete problem writes:

\[
\int_{\Omega} \frac{1}{\Delta t} \mathbf{u}_h^{n+1} \cdot \mathbf{\varphi}_h + a(\mathbf{u}_h^{n+1}, \mathbf{\varphi}_h) + b(\mathbf{\varphi}_h, p_h^{n+1}) = L(\mathbf{\varphi}_h) + \int_{\Omega} \frac{1}{\Delta t} \mathbf{u}_h^n \cdot \mathbf{\varphi}_h \quad \forall \mathbf{\varphi}_h \in \mathbf{V}_h \\
\]

(71)

\[
b(\mathbf{u}_h^{n+1}, q_h) = 0 \quad \forall q_h \in Q_h
\]

(72)

We choose the following compatible spaces (see [15]):

- \( \mathbf{V}_h = \{ \mathbf{v} \in C^0(\Omega) \text{ s.t. } \mathbf{v}|_K \in P_2^k \text{ for all } K \in \mathcal{T}_h \} \)
- \( Q_h = \{ q \in C^0(\Omega) \text{ s.t. } q|_K \in P_1 \text{ for all } K \in \mathcal{T}_h \} \cap L_0^2(\Omega) \)

Where \( \mathcal{T}_h \) is the set of mesh elements and \( P_k \) is the set of polynomial function of degree \( k \). The space \( \mathbf{V}_h \) can be adjust for specific boundary condition.
We denote by $U^n$, $P^n$ the vectors

$$U^n = \begin{pmatrix} \alpha_1^n \\ \vdots \\ \alpha_N^n \end{pmatrix}, \quad P^n = \begin{pmatrix} \beta_1^n \\ \vdots \\ \beta_M^n \end{pmatrix}$$

(73)

A the matrix $A_{i,j} = a(\varphi_i, \varphi_j)$ for $1 \leq i, j \leq N$ and $B$ the divergence matrix $B_{i,j}^T = b(\varphi_i, \psi_j)$ for $1 \leq i \leq N, 1 \leq j \leq M$. We also note $F = (\gamma_i)^T$ where $f = \sum_{i=1}^N \gamma_i \varphi_i$ and $M$ the mass matrix.

The problem writes:

$$
\begin{pmatrix}
A + M/\Delta t & B^T \\
B & 0
\end{pmatrix}
\begin{pmatrix}
U_{n+1} \\
P_{n+1}
\end{pmatrix}
= 
\begin{pmatrix}
F \\
0
\end{pmatrix}
+ 
\begin{pmatrix}
M/\Delta t & 0 \\
0 & 0
\end{pmatrix}
\begin{pmatrix}
U^n \\
0
\end{pmatrix}
$$

(74)

4.3. Stokes-Exner coupling

The algorithm for the Navier-Stokes equations constitute the main part of the complete method. The coupled algorithm can be summarized by the following:

**Algorithm 1 Stokes-Exner coupling**

Require: $\Omega^0$, $b^0$, $w^0$, $u^0$

for $n = 0$ to $N_T - 1$ do

$u^{n+1} = \text{StokesSolver}(\Omega^n, u^n, w^n)$

$b^{n+1} = \text{ExnerSolver}(b^n, u^{n+1})$

$d_t^{n+1} = b^{n+1} - b^n$

$d_{n+1} = \text{ALESolver}(\Omega^n, d_t^{n+1})$

$w^{n+1} = d^{n+1} - d^n$

$\Omega^{n+1} = \Omega^n + d_t^{n+1}$

end for

5. Numerical tests

5.1. Analytical test for the Stokes problem

In order to validate our implementation of the Stokes problem, we use the known solution of Bercovier-Engelman (REF):

$$v = \begin{pmatrix}
-256y(y-1)(2y-1)x^2(x-1)^2 \\
256x(x-1)(2x-1)y^2(y-1)^2
\end{pmatrix}

p = (x-0.5)(y-0.5)

f = \begin{pmatrix}
256(x^2(x-1)^2(12y-6) + y(y-1)(2y-1)(12x^2-12x+2)) + (y-0.5) \\
-256(y^2(y-1)^2(12x-6) + x(x-1)(2x-1)(12y^2-12y+2)) + (x-0.5)
\end{pmatrix}
$$

We can compare the exact solution and the approximation in Fig 3.
5.2. Driven cavity

We also study the boundary condition see in section ??, For this, we want to reproduce a driven cavity. It means that we have $v|_{\Gamma_{in} \cup \Gamma_{out} \cup \Gamma_b} = 0$ and $v|_{\Gamma_s} = (1, 0)^T$. In Fig 4, you can see the expected behavior of the velocity.

5.3. Fluvial dune test case

It is not straightforward to validate the method on the complete model with an analytical solution. It this part, the objective is to evaluate the coupling model with the classical test case of the dune migration. If we initialize the simulation with a succession of dunes and a fluvial flow, it is expected that the dunes migrate in the same direction as the flow.
6. Conclusion

In this note, we consider a coupling between the Exner equations and the Stokes equation to model the transport sediments in flow phenomena. We focus on a model without free surface and used some numerical test to evaluate the relevance of the method. The fluid structure interaction theory and method have been applied for the simplified and the objective is to test the proposed method which can be extend to a free surface model. The library Feel++ and the high computing performance embedded have been used to test the solution method. Therefore, the final goal of this project is to understand the impact of the sediment transport on the flow using Navier-Stokes with a free surface system coupled with the standard Exner equation.

7. Acknowledgments

A completer We acknowledge the GDR EGRIN for its financial support and the CIRM for its warm welcome during the CEMRACS.

8. Annexe: Implementation in Feel++ framework for the Stokes Problem

Feel++ is a C++ library for partial differential equation solves using generalized Galerkin methods such as the finite element method, the h/p finite element method, the spectral element method or the reduced basis method. The source code can be found here. We first need to define the type of the mesh that we’ll use. For this, we need to precise the dimension and the type of elements used, simplex or hypercube. This is done with the lines:

```cpp
typedef Simplex<2> convex_type;
typedef Mesh<convex_type> mesh_type;
typedef boost::shared_ptr<mesh_type> mesh_ptrtype;
```

The last line creates a pointer to a mesh.

Next, we have to precise the function space, from section ?? we know that we have to use Taylor-Hood function space. That is $P_2^2 \times P_1$. We use the following lines to do so:

```cpp
typedef Lagrange<2, Vectorial> base_u; // $[P2]^2$
typedef Lagrange<1> base_p; // $P1$
typedef FunctionSpace<mesh_type, bases<base_u, base_p>> space_type;
typedef boost::shared_ptr<space_type> space_ptrtype;
```

We now have the needed types, we can use them to define the problem. First, we load the geometry by using the fonction loadMesh. This function takes several parameters, but most of them have a default value that we won’t have to change. In particular, the parameter name define which file to use, it is set by default to the value of the option gmsh.filename. Another mandatory parameter is the type of the mesh that we have defined previously.

```cpp
auto mesh = loadMesh(_mesh=new mesh_type);
```

The keyword auto has been introduced in C++11, and permits to infer the type of a variable with the declaration. Since Feel++ has a intensive usage of the templates, the use of the auto keyword makes the code clearer. On the other hand, this hides the real type of the variables. Next, we declare the function space used. So we use the type previously defined. We can then get an element of this space by using the element method. At this point, we have an element $(\mathbf{v}, p) \in P_2^2 \times P_1$, and we can get each element of the tuple with the following lines:
We can use the same method to define the test functions.

We’re almost ready to define the variational formulation, but first we need to set the functions used as right hand side and boundary condition. We use the option to set them, this will allow us to change those functions at the execution, without the need to recompile the application.

```cpp
auto f = expr<2,1>(soption("functions.f"));
auto g = expr<2,1>(soption("functions.g"));
```

What is happening here, is that we use the option `functions.f` and `functions.g` to define the functions used. The method `soption` transforms these options as a string (you have also the methods `doption`, `ioption` and `boption` to transform an option respectively as a double, an integer and a boolean).

The function `expr` will transform the string taken as parameter into an expression usable within Feel++. The templates parameters are the size of the expression, in this case, a matrix with 2 lines and 1 column, that is a 2D vector (in Feel++, the vectors are column vectors).

We can now define our variational formulation. We first set the linear form $\int_\Omega f \cdot \varphi$.

```cpp
auto L = form1(_test=VhQh);
L = integrate(_range=elements(mesh),
             _expr=inner(f,id(phi)));
```

The first line says that the linear form operates on the composite space $P_2^2 \times P_1$. The second line, that $L$ will be the integrale on all the elements of the mesh of the expression $f \cdot \varphi$.

The bilinear form operates on $(P_2^2 \times P_1) \times (P_2^2 \times P_1)$. Where one of them is the space where lives the test functions, and the other one is where lives the solution, it is called the trial space.

The difference between these two spaces is made by the operators in the integrals. While the test functions have the usual operators, the trial ones use the operators with a suffix $t$. The list of the operators can be found here. Another useful suffix is $v$, it is the evaluation of a function.

We use the formulation described in section ??, $\int_\Omega \nabla v : \nabla \varphi - p \nabla \cdot \varphi - q \nabla \cdot v$.

```cpp
auto A = form2(_trial=VhQh, _test=VhQh);
A = integrate(_range=elements(mesh),
             _expr=inner(gradt(v),grad(phi)));
A += integrate(_range=elements(mesh),
               _expr=-idt(p)*div(phi) - id(q)*divt(v));
```

Regarding the boundary conditions, we apply them strongly by the use of the `on` method of the bilinear form. This method takes as parameters the range on which apply the boundary condition, that is $\Gamma$. It takes also the right hand side associated with the bilinear form, and the element on which to apply those boundary condition. Finally, it takes the expression of the boundary condition.

```cpp
A += on(_range=boundaryfaces(mesh), _rhs=L, _element=v, _expr=g);
```

We can change the range of the boundary, if we want to apply them only on $\Gamma_{in}$, we use `markedfaces(mesh, "in")` instead of `boundaryfaces(mesh)`. It is important however that the geometry has the marker “in” somewhere.
Finally, we can solve the system, simply by calling the method `solve` on the bilinear form, with as parameters, the right hand side and the element representing the solution.

```plaintext
A.solve(_rhs=L, _solution=VP)
```

The solver and preconditioner used to solve the system are implemented by PETSc. You can see some informations about them with respectively the `ksp-view` and `pc-view` options. You can then change what is used with the `ksp-type` and `pc-type` options.

### References


